executable code and be more directed to generating analysis information such as cross-reference data, object class graphical hierarchies, function call relationships and other program visualization tool data. In such instances, the source code analysis may be less concerned with detecting, reporting and resolving errors. Although not preferred, some cross-reference information may be obtained through simple scanning, with little reference to grammar-driven analysis. An exemplary method of parsing in this way is described in Applicant's co-pending United States Patent Application entitled "METHOD AND TECHNIQUE FOR APPROXIMATE GENERATION OF SOURCE CODE CROSS-REFERENCE INFORMATION", filed 7/11/02 under Serial No. 10/192,596 and incorporated herein by reference.

[0049] As the parse of a source code file progresses, global line reference and local line reference information are generated for each source code identifier encountered (step 310) by cross-reference generator 140 (Fig. 1). In a parsing method in which a single pass is made through a source code file, it is not possible to type resolve all source code identifiers. Thus, some file-reference information generated does not have sufficient type and namespace information to uniquely describe the identifier encountered. Step 312 performs additional type resolution on the identifiers in the cross-reference information using the knowledge of the identifiers gleaned from the complete parse to clean-up any cross-references generated by the parser before the type information was determined. As per well-known parsing techniques, symbol tables may be constructed during parsing for each source file and any included files to manage the determination and type resolution of the identifiers. It is understood that some